Scholia 2026: Compliance with SPARQL 1.1

Egon L. Willighagen[1], Daniel Mietchen[2,3,4], Peter Patel-Schneider, Konrad Linden[5], Johannes Kalmbach[5], Lars G. Willighagen[6], Wolfgang Fahl[7] and Hannah Bast[5]

[1]*Department of Translational Genomics, NUTRIM, Maastricht University, The Netherlands*
[2]*FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Karlsruhe, Germany*
[3]*Leibniz Institute of Freshwater Ecology and Inland Fisheries (IGB), Berlin, Germany*
[4]*Institute for Globally Distributed Open Research and Education (IGDORE), Jena, Germany*
[5]*Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany*
[6]*Department of Ecology, Radboud Institute for Biological and Environmental Sciences, Radboud University, The Netherlands*
[7]*BITPlan GmbH, Willich, Germany*

Abstract

Scholia is a specialized portal that serves Wikidata content through a set of 387 precomposed SPARQL query templates. Scholia used to query the Wikidata Query Service (WDQS), powered by the Blazegraph SPARQL engine. This service has been facing major efficiency challenges for years, which eventually led to what is known as the *WDQS graph split*. As a consequence of this graph split, there was no longer a single service for the complete data, but two separate services for about half of the data each. Since Scholia queries make extensive use of the complete data, a new solution was needed. In this demonstration paper, we describe how Scholia was migrated to the QLever SPARQL engine. As part of this process, all Blazegraph-specific constructs were removed and all queries were rewritten to be fully compliant with the SPARQL 1.1 standard. At the same time, performance was significantly improved.

Keywords

Scholia, Wikidata, SPARQL, Python

Wikidata is a crowd-curated platform providing general knowledge and references in a semantic format [1]. Its official SPARQL service, the Wikidata Query Service (WDQS), has historically relied on the Blazegraph engine to evaluate queries [2]. However, as Wikidata grew to over 17 billion triples, its Blazegraph backend has reached significant performance limits. This led the Wikimedia Foundation to split the query service into two parts in 2025: *WDQS-main*, containing 8.6 billion triples related to general knowledge, and *WDQS-scholarly*, containing 8.7 billion triples related to scholarly articles [3].

Scholia [4] is a specialized portal for the scientific community that serves Wikidata content through a set of 387 precomposed SPARQL query templates organized into over 40 profile types, called *aspects*. These cover general entities such as authors, works, venues, organizations, events, and event series, as well as more special ones like taxa, diseases, chemicals, genes or proteins [5]. Because many Scholia queries require data from both WDQS-main and WDQS-scholarly, the split posed a critical challenge to the continuation of the service. As an interim solution, the Wikimedia Foundation continued to provide a Blazegraph instance with the unified full graph until 20 January 2026. Preparing for this change, the Scholia team investigated three primary recovery routes, ultimately selecting the third one:

1. Modifying existing queries to use SPARQL federation to communicate between the two endpoints. This was found to be impractical, as inter-service communication led to severe performance degradation and the loss of optimization opportunities [3].
2. Using snapQuery middleware [6] to generalize query access across endpoints.
3. Remove all Blazegraph-specific constructs in Scholia and make all queries fully compliant with the SPARQL 1.1 standard. This allows the use of any SPARQL 1.1 engine that is efficient enough to process queries on the complete (unsplit) Wikidata graph.

SWAT4HCLS 2026: The 17th International Conference on Semantic Web Applications and Tools for Health Care and Life Sciences, March 23–26, 2026, Amsterdam, The Netherlands
egon.willighagen@maastrichtuniversity.nl (E. L. Willighagen)
0000-0001-7542-0286 (E. L. Willighagen); 0000-0001-9488-1870 (D. Mietchen); 0009-0000-7789-7459 (P. Patel-Schneider); 0009-0009-1449-5423 (K. Linden); 0000-0002-5582-161 (J. Kalmbach); 0000-0002-4751-4637 (L. G. Willighagen); 0000-0002-0821-6995 (W. Fahl); 0000-0003-1213-6776 (H. Bast)

A fork of the original Scholia project was created [7] to collaborate on the necessary modifications, around which several hackathons were organized [8]. A decision was made to use a Wikidata query service (https://qlever.dev/api/wikidata) powered by QLever [9, 10]. This decision was supported by benchmarking multiple SPARQL engines [11], of which QLever was fastest overall.

The bulk of the work was eliminating non-standard constructs from query templates. The major types of these constructs were:

- *named sub-queries* to give names to sub-patterns, for multiple reuse within a query;
- the Blazegraph *locality service*, for finding located entities within a given radius of a given point;
- the Blazegraph *gather-scatter service*; for finding nodes reachable from a given node via k hops;
- the *MediaWiki API service* for accessing contents from the Wikipedias (e.g., via full-text search);
- the Wikibase *label service* for finding labels for items, for a given sequence of fallback languages.

Most of these constructs originated as workarounds for efficiency problems with Blazegraph. For the same reason, they are also used by many other systems using the Wikidata query service. An engine like QLever, that can efficiently handle standard SPARQL constructs at the scale of Wikidata, obviates the need for such workarounds.

Specifically, named sub-queries were eliminated by substituting the named query where it was referenced. The locality service was eliminated by unrestricted querying and filtering based on computed distances. The gather-scatter service was eliminated by using transitive closure and filters or limits.

The hardest construct to eliminate was the Wikibase label service, because it depends on access to the entire query to determine the relevant variables and because there are many labels in Wikidata. We solved this by using a macro that takes the variables as an argument and is expanded before the query is sent for evaluation. The expansion uses a sequence of *OPTIONAL* joins based on the HTTP *Accept-Language* header sent by the browser, with a preceding *BIND* to handle potentially unbound variables that can come from previous *OPTIONAL* constructs in the query.

The process took about eighteen months, with the results we had hoped for. In particular, all Scholia queries are now compliant with the SPARQL 1.1 standard, and at the same time, using QLever, most queries are now significantly faster than with the original setup using Blazegraph.

Since the Scholia queries are complex and cover a wide range of SPARQL constructs, our effort provides strong evidence that it is realistic to make other Wikidata-based systems fully compliant with the SPARQL 1.1 standard as well, and that QLever can handle the associated efficiency challenges. In particular, this is true for the WDQS itself, which the Wikimedia Foundation plans to migrate off Blazegraph. We have not investigated whether other SPARQL-compliant engines can handle these challenges, too, and to which degree further query rewriting would be needed for them.

The demo at SWAT4HCLS will show query updates of various Scholia aspects to exemplify the changes made and the performance improvements achieved by using QLever. This includes a demo the Scholia page for the SWAT4HCLS event series [12]

Acknowledgments

The authors thank everyone who has contributed patches or bug reports to the Scholia project, particularly Finn Nielsen (for the original concept of Scholia), Moritz Schubotz and Lane Rasberry. A rich list of contributors to the project is available at https://github.com/WDscholia/scholia/graphs/contributors. The work described here was supported in part by the German Research Foundation through the find.software project [13] (funded under grant number 567156310) and by the European Commission through the AQUANAVI initiative [14] of the OSCARS project (101129751).

Declaration on Generative AI

Some author(s) have occasionally employed Generative AI tools for the changes in the Scholia code. All code changes have been reviewed by another person.

References

[1] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Communications of the ACM 57 (2014) 78–85. doi:10.1145/2629489.

[2] S. Malyshev, M. Krötzsch, L. González, J. Gonsior, A. Bielefeldt, Getting the most out of Wikidata: Semantic technology usage in Wikipedia's knowledge graph, in: International Semantic Web Conference, Springer, 2018, pp. 376–394.

[3] T. Lubiana, L. Rasberry, D. Mietchen, The Wikidata Query Service Split and its Impact on the Scholarly Graph, in: CEUR Workshop Proceedings, volume 4064, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-4064/PD-paper3.pdf.

[4] F. Å. Nielsen, D. Mietchen, E. Willighagen, Scholia, Scientometrics and Wikidata, in: The Semantic Web: ESWC 2017 Satellite Events, ESWC 2017 Satellite Events, Springer International Publishing, Portorož, SI, 2017, pp. 237–259. doi:10.1007/978-3-319-70407-4_36.

[5] E. Willighagen, D. Slenter, A. Rutz, D. Mietchen, F. Å. Nielsen, Scholia Chemistry: access to chemistry in Wikidata (2025). doi:10.26434/CHEMRXIV-2025-53N0W.

[6] W. Fahl, T. Holzheim, C. T. Hoyt, D. Priskorn, E. Willighagen, SnapQuery, 2025. URL: https://github.com/WolfgangFahl/snapquery.

[7] F. Å. Nielsen, D. Mietchen, E. Willighagen, H. Bast, Scholia, 2017. URL: https://github.com/ad-freiburg/scholia/tree/qlever, GitHub repository qlever branch.

[8] Wikidata, Scholia Events December 2025, 2025. URL: https://www.wikidata.org/wiki/Wikidata:Scholia/Events/2025_12.

[9] H. Bast, B. Buchhold, QLever: A Query Engine for Efficient SPARQL+Text Search, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 647–656. doi:10.1145/3132847.3132921.

[10] H. Bast, J. Kalmbach, T. Klumpp, F. Kramer, N. Schnelle, Efficient and Effective SPARQL Autocompletion on Very Large Knowledge Graphs, in: CIKM '22: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Association for Computing Machinery, Atlanta, GA, USA, 2022, pp. 2893–2902. URL: http://dx.doi.org/10.1145/3511808.3557093. doi:10.1145/3511808.3557093.

[11] P. F. Patel-Schneider, Benchmarking SPARQL Engines on Wikidata Queries, in: CEUR Workshop Proceedings, volume 4108, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-4108/paper3.pdf.

[12] Event Series - Semantic Web Applications and Tools for Healthcare and Life Sciences, Scholia, 2026. URL: https://qlever.scholia.wiki/event-series/Q56846035, accessed: 2026-02-08.

[13] R. Gey, D. Mietchen, O. Karras, T. Wittenborg, M. Schubotz, J. Bumberger, find.software: Foundations for Interdisciplinary Discovery of (Research) Software, Research Ideas and Outcomes 11 (2025) e179253. doi:10.3897/rio.11.e179253.

[14] T. Heger, S. Angela Berger, J. M. Jeschke, C. Kittel, P. Kraker, A. K. Makower, D. Mietchen, J. C. Nejstgaard, M. Schramm, AQUANAVI: Navigating Grand Challenges and their Mitigation using Aquatic Experimental RIs, Research Ideas and Outcomes 11 (2025) e176476. doi:10.3897/rio.11.e176476.